Research Article

An Efficient Mesh Generation Method for Fractured Network System Based on Dynamic Grid Deformation

Shuli Sun, Jie Sui, Bin Chen, and Mingwu Yuan

LTCS, Department of Mechanics and Engineering Sciences, College of Engineering, Peking University, Beijing 100871, China

Correspondence should be addressed to Shuli Sun; sunsl@mech.pku.edu.cn

Received 18 July 2013; Accepted 8 August 2013

Academic Editor: Song Cen

Copyright © 2013 Shuli Sun et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Meshing quality of the discrete model influences the accuracy, convergence, and efficiency of the solution for fractured network system in geological problem. However, modeling and meshing of such a fractured network system are usually tedious and difficult due to geometric complexity of the computational domain induced by existence and extension of fractures. The traditional meshing method to deal with fractures usually involves boundary recovery operation based on topological transformation, which relies on many complicated techniques and skills. This paper presents an alternative and efficient approach for meshing fractured network system. The method firstly presets points on fractures and then performs Delaunay triangulation to obtain preliminary mesh by point-by-point centroid insertion algorithm. Then the fractures are exactly recovered by local correction with revised dynamic grid deformation approach. Smoothing algorithm is finally applied to improve the quality of mesh. The proposed approach is efficient, easy to implement, and applicable to the cases of initial existing fractures and extension of fractures. The method is successfully applied to modeling of two- and three-dimensional discrete fractured network (DFN) system in geological problems to demonstrate its effectiveness and high efficiency.

1. Introduction

The progress of computer hardware and software techniques has made it possible to analyze large-scale engineering problems in detail by numerical solution, for example, the finite element method. The key to success of the method depends very much on proper discretization or meshing, especially for those problems with complicated three-dimensional geometric domain. The quality of mesh has been known to affect significantly both the efficiency and the accuracy of the numerical solution [1]; therefore, to achieve high quality mesh is always a goal for researchers. This paper focuses on initial mesh generation and remeshing during deformation in numerical simulation of fractured network system in geological problems.

Modeling and simulation of geological problems are usually tedious and time consuming. DFNs have a rather complex geometry, especially in three-dimensional case. Initial fractures exist in such fractured network system, and new fractures will be created during deformation of geological

media. The existence of fractures, especially the extension of the fractures, brings lots of troubles in mesh generation of DFNs. Geometric constraints induced by fractures significantly increase degree of difficulty of meshing algorithm. Another problem that must be faced is the efficiency of meshing algorithm. Usually, a model of fractured network system in practical engineering problems may include many fractures. If more than 30 fractures are considered in three dimensional case, a hundred million elements of mesh may be needed to work on.

Many methods are developed to discretize complex fractured media; however, due to the complexity of the problem, most of them have been used for simple fractured configurations [2]. For example, Blesent et al. [3] developed a new method to discretize nonplanar fractures. Their approach generates fine tetrahedrons close to fractures and identifies certain tetrahedron faces as fracture faces. That method efficiently approximates single-fracture configuration. However, the method becomes computationally costly and difficult to implement for complex fractured media [2].

To exactly construct fracture boundaries, the so-called boundary recovery operation based on topological transformation [4–7] is usually involved. This operation relies on many complicated skills and is not easy to implement. This paper presents an alternative and efficient approach for meshing the fractured network system. The method firstly presets points on fractures, and performs Delaunay triangulation to obtain preliminary mesh by point-by-point centroid insertion algorithm. The fractures are then exactly recovered by local correction with dynamic grid deformation approach. The proposed method is efficient, easy to implement, and applicable to the cases of initial existing fractures and extension of fractures. It can exactly construct fracture boundaries without doing complicated topological transformation operations.

In this paper, point-by-point centroid insertion algorithm for Delaunay triangulation and generation of preliminary mesh are given in Section 2. Then in Section 3, an efficient local correction approach based on revised dynamic grid deformation is presented to exactly recover fracture boundaries. A smoothing procedure by error-based quality metric is developed to improve the quality of mesh in Section 4. Two- and three-dimensional examples are given to illustrate performance of the proposed method in Section 5. Finally in Section 6, it is concluded that the approach presented in this paper is applicable to modeling of two- and three-dimensional discrete fractured network (DFN) system in geological problems for both cases of initial existing fractures and extension of fractures with high efficiency.

2. Generation of Preliminary Mesh

Complexity of meshing for fractured network system is coming from geometric constraints induced by fracture boundaries. The so-called boundary recovery operation based on topological transformation is usually involved in exact recovery of the fracture boundaries. This operation relies on many complicated skills and is not easy to implement. In this paper we present an alternative way to recover the fracture boundaries. Preliminary mesh is firstly generated by point-by-point centroid insertion algorithm without considering boundary recovery. Then the fracture boundaries are exactly recovered by local correction with revised dynamic grid deformation approach. To avoid the topological transformation will raise the efficiency of meshing tremendously. This section firstly introduces point-by-point centroid insertion algorithm.

2.1. Point-by-Point Centroid Insertion Algorithm [8]. There exist many triangulation possibilities for a given set of points. Among them, the so-called Delaunay triangulation is the most commonly used unstructured triangulation due to its large class of well-defined properties, such as the well-known empty circumcircle or circumsphere property. Point-insertion algorithms [9] can be employed to construct Delaunay triangulation. An initial triangulation is firstly constructed to cover the whole region to be meshed. Then mesh points are put in a list and inserted sequentially into the existing triangulation using the Bowyer-Watson algorithm

[10, 11] or Lawson algorithm [12]. The final mesh is obtained when all points from the list have been inserted.

Many works [13–15] have been done to raise the efficiency of the algorithm. In this paper, the point-by-point centroid insertion algorithm [8], which is one of the fastest triangulation algorithms known by authors, is adopted to generate the preliminary mesh due to its efficiency and robustness. A group of double link data structure corresponding to element inserting coefficient (EIC) is developed to eliminate the searching for the element with maximum EIC in the point-by-point centroid insertion algorithm. Adjacency search, double linked element list, random direction search, adjacency rotation, and heredity of geometrical quantities are applied to increase the efficiency of mesh generation. The main steps of the algorithm are summarized as follows:

- (a) generate points on boundary surface of the domain;
- (b) construct an initial triangulation to completely cover the domain;
- (c) insert points on boundary surface by Bowyer-Watson algorithm;
- (d) delete topological nonconforming elements;
- (e) insert centroid of element by Bowyer-Watson algorithm until EIC for every element is zero.

2.2. Presetting Points on Fracture Boundaries. The geometric data of fractures must be analyzed and modified if necessary to ensure the applicability for mesh generation and the finite element simulation [16]. Some tiny fractures are removed or merged into the big ones. Then points along fracture boundaries are inserted in advance according to predefined element size. Thus the fractures are confined by these presetting points and described or constructed in some sense (see Figure 1 for 2D illustration).

2.3. Generation of Preliminary Mesh. With the retained all predefined points on fractures, the next step is to generate the preliminary mesh for whole computational region by point-by-point centroid insertion algorithm. The points on outer boundaries and the preset points on fractures are firstly inserted (see Figure 2 for 2D illustration). Then continue to insert centroid of element successively by the algorithm introduced in Section 2.1 to generate the so-called preliminary mesh (see Figure 3). In this process we do not consider the recovery of the boundary to maintain fractures; thus the meshing algorithm is rather simple to implement and remains of high efficient performance.

3. Local Correction Approach by Dynamic Grid Deformation

Generally, the above approach cannot guarantee the geometric integrity for each fracture boundary. In Figure 3 we can see that the fracture is not exactly constructed. Such undesired configuration may happen quite often in random three-dimensional DFN systems. It is mandatory to perform geometric local correction, which modifies slightly the boundary

FIGURE 1: Presetting points on fracture.

FIGURE 2: Delaunay triangulation by point-by-point insertion.

of the fractures. In this section, a local correction approach based on revised dynamic grid deformation to exactly recover fracture boundaries is presented, which searches approximate fracture boundary first.

3.1. Searching for Approximate Fracture Boundaries. Consider a 2D fracture boundary P_1 - P_2 shown in Figure 4(a); the procedure to search its approximate fracture boundary (see Figure 4(b)) consists of 5 main steps described as follows.

- (1) Construct the list of adjacent nodes (AP-List) for all the nodes in the preliminary mesh. For example, the AP-List corresponding to node P in Figure 4(c) consists of all the black nodes around P .
- (2) Denote all the nodes on approximate fractural boundary as AFB-List. Add P_1 into AFB-List, and let $P = P_1$.
- (3) If the AP-List of node P contains P_2 : add P_2 to the AFB-List, end searching, go to (5); else, go to (4).
- (4) Search a node S in AP-List for P , if node S satisfies the following three conditions: add S into AFB-List, let $P = S$, and go to (3).
 - (a) Consider $\vec{PS} \cdot \vec{PP_2} > 0$.
 - (b) Nodes S , P , and V are not in the same triangle element, where V is the previously included node in AFB-List just in front of P .
 - (c) Node S is nearer to line P_1 - P_2 than any other node which is in AP-List and satisfies (a) and (b).
- (5) All the nodes in AFB-List composed a polygonal line which can be treated as the approximate fracture boundary.

FIGURE 3: Preliminary mesh without recovery of fracture boundary.

After finding the approximate fracture boundaries (see Figure 4(b)), a local correction approach by revised dynamic grid deformation can be employed to exactly recover fracture boundaries. The following subsection gives the dynamic grid deformation algorithm which is originally proposed by Liu et al. [17].

3.2. Dynamic Grid Deformation Algorithm. Dynamic grid algorithm [17–24] is one of the key techniques in numerical simulation of fluid-structure interaction problems, where the mesh has to dynamically change to cover the changed solution domain. Due to the efficiency, the mesh deformation approach is one of the most desirable categories of the dynamic grid method.

The widely used mesh deformation methods include spring analogy [18, 19] and elastic solid based methods [20, 21]. Liu et al. [17] proposed a simple and efficient method for dynamic grid deformation. This method is based on Delaunay graph mapping of the original mesh. A Delaunay graph of the solution domain is firstly generated. The geometric movement is carried out using the Delaunay graph with ease and efficiency. The original grid is then mapped back onto the deformed Delaunay graph to provide the new mesh. The main steps of the method are summarized as follows:

- (a) generate the Delaunay graph of the original mesh by outer and inner boundary points;
- (b) locate the mesh points in the Delaunay graph, and calculate the relative area/volume ratio coefficients (actually the area/volume coordinates) for each mesh point;
- (c) move the Delaunay graph according to the specified boundary change;
- (d) relocate/map the mesh points in the moved Delaunay graph according to the relative area/volume ratio coefficients.

3.3. Local Correction by Revised Dynamic Grid Deformation. The above noniterative algebraic approach is easy to implement and works for any type of mesh. In the whole

FIGURE 4: Searching approximate fracture boundaries for a real fracture P_1 - P_2 .

procedure, most calculations concentrate in steps (b) and (d). Facing to the most time consuming parts of Liu's approach (mesh points locating and mapping/relocating), Sun et al. [24] presented a high efficient algorithm and implementation scheme to speed up the method. Two main improvements were proposed. First, a fast locating technique was developed to locate the background element for the mesh points. Second, an efficient scheme was proposed that avoided most of repeated calculations in relocation of the mesh points in the graph. Both time complexity analysis and testing results indicated that substantial speedup is achieved for the proposed algorithm and implementation, while the memory requirement is even decreased.

In preliminary mesh (Figure 3) it can be seen that some points on the approximate fracture boundaries are slightly apart from the exact boundary of the fractures. If the distances of these points to the exact boundary are regarded as movement of inner boundaries, then the dynamic grid deformation algorithm can be employed to recover the exact fracture boundaries. This is the most important idea of the proposed method. The points which are already on the exact boundaries are allowed to remain fixed or move slightly along the boundary.

In the dynamic grid deformation algorithm proposed by Liu et al. [17], the inner boundaries are usually closed. However, the inner fracture boundaries (see, e.g., Figure 3), are unclosed and intersected. It is difficult to maintain the approximate boundaries when generating Delaunay graph in this case. Fortunately it is not necessary to maintain

the approximate boundaries in our algorithm. Only the preset points on fracture boundaries and key points on outer boundaries are required to generate Delaunay graph. Thus we have the following revised dynamic grid deformation algorithm based on [24] to perform local correction for approximate fracture boundaries:

- (a) search for approximate fracture boundaries (see Figure 4(b) for 2D illustration);
- (b) generate the Delaunay graph by all points on outer and approximate boundaries (see Figure 5(a) for 2D and 5(b) for 3D);
- (c) locate the mesh points in the Delaunay graph, calculate and save some intermediate parameters rather than the relative area/volume ratio coefficients [24];
- (d) compute the movement of points on approximate fracture boundaries;
- (e) move the Delaunay graph according to the above movement (see Figures 5(c) and 5(d));
- (f) relocate/map the mesh points in the moved Delaunay graph according to the parameters saved in step (c) [24], and obtain the final meshes (see Figure 6).

The movement of Delaunay graph is demonstrated in Figure 5, and the final meshes are given in Figure 6.

The proposed method is efficient, easy to implement, and applicable to the cases of initial existing fractures and extension of fractures. It can exactly construct fracture boundaries

FIGURE 5: Delaunay graph before and after movement. (a) Initial Delaunay graph (2D); (b) initial Delaunay graph (3D); (c) Delaunay graph after movement (2D); (d) Delaunay graph after movement (3D).

without performing complicated topological transformation operations.

4. Mesh Quality Improvement

After the local corrections, the fracture boundaries are well recovered; however, the quality of mesh in particular local region is probably unsatisfactory. Performing extra mesh quality improvement is a good choice. In this paper a smoothing algorithm which takes care of the existence of points on fracture boundaries is employed to raise the mesh quality. Theoretically the optimization-based smoothing procedures [25–27] would lead to better results. Unfortunately, application of such kind of procedure in practical problem is limited

due to its time consuming cost. In many works, smoothing was commonly performed by Laplacian smoothing technique, that is, simply shifting each interior node (free vertex) to the centroid of the surrounding polyhedron. Although computationally inexpensive and easy to implement, this method may produce invalid or illegal tetrahedrons occasionally [27] that are unacceptable in numerical analysis.

Chen and Xu presented the concept of optimal Delaunay triangulation [28] and developed a quality smoothing scheme for triangular mesh by minimizing the interpolation error among all triangulations with the same number of vertices in the local patch [29]. This optimization-based smoothing method could solve the optimization problem exactly, and thus the computational cost is as low as that of Laplacian

FIGURE 6: Final mesh with exact fracture boundary. (a) 2D mesh; (b) cross-section of 3D mesh.

smoothing. Combining this smoothing scheme and the concept of optimal Delaunay triangulation, Alliez et al. [30] proposed a Delaunay-based variational approach for tetrahedral meshing by minimizing a simple mesh-dependent energy through global updates of both vertex positions and connectivity, which is very effective to improve the quality of “bad” tetrahedrons. Based on the above algorithms, a modified smoothing scheme is proposed by Sun et al. [31] to avoid possibility of producing illegal elements while maintaining high efficiency of the above smoothing schemes. The main steps of the scheme for three-dimensional case are listed as follows [31].

- (a) Calculate the optimal position x_i^* for free vertex x_i by

$$x_i^* = -\frac{1}{2|\Omega_i|} \sum_{T_j \in \Omega_i} \left(\nabla_{x_i} |T_j| \left(\sum_{\substack{x_k \in T_j \\ x_k \neq x_i}} \|x_k\|^2 \right) \right), \quad (1)$$

where Ω_i denotes the first ring of vertex x_i , $|T_j|$ is the volume of tetrahedron T_j with respect to vertex x_i , and $|\Omega_i|$ denotes the volume of Ω_i .

- (b) Check if x_i^* causes illegal element.
(c) If there exists no illegal element, accept the new position; otherwise, solve the optimization problem in (2) with constraints to obtain the new position of x_i :

$$\begin{aligned} \min \quad & E_{\Omega_i}(x_i) = \frac{1}{4} \sum_{T_j \in \Omega_i} \left(|T_j| \sum_{\substack{x_k \in T_j \\ x_k \neq x_i}} \|x_k\|^2 \right) + \frac{|\Omega_i|}{4} \|x_i\|^2 \\ \text{s.t.} \quad & |T_j| > 0. \end{aligned} \quad (2)$$

The optimization problem in (2) can be solved by integrating chaos search and BFGS algorithm efficiently.

Considering the existence of points on fracture boundaries, this section presents a revised smoothing scheme to improve the quality of mesh. The algorithm is modified as follows.

- (1) For vertex x_i on fracture boundaries, solve the optimization problem in (3) for obtaining the new position of x_i with the constraint equation $b(x_i) = 0$, which means that x_i cannot be apart from the fracture boundary. Consider

$$\begin{aligned} \min \quad & E_{\Omega_i}(x_i) = \frac{1}{4} \sum_{T_j \in \Omega_i} \left(|T_j| \sum_{\substack{x_k \in T_j \\ x_k \neq x_i}} \|x_k\|^2 \right) + \frac{|\Omega_i|}{4} \|x_i\|^2 \\ \text{s.t.} \quad & |T_j| > 0, \quad b(x_i) = 0. \end{aligned} \quad (3)$$

- (2) For other free vertex x_i , employ steps (a) to (c) for its optimal position.

5. Testing Examples

Two examples are selected to demonstrate the effectiveness and efficiency of the proposed approach. Both examples are tested on Windows 7 system with Intel (R) Core (TM) i3-2120 CPU (3.30 GHz) and 16 GB memory. The first example is a two-dimensional fracture network system with 13 individual and intersected fractures (lines in Figure 7(a)). By the proposed techniques of inserting predefined nodes on fractures and local correction, the fractures are constructed precisely. The final mesh includes 1154 nodes and 2216 triangles. The

FIGURE 7: Mesh of 2D fractured network system. (a) A 2D fracture network system; (b) preliminary mesh with approximate fracture boundaries; (c) final mesh with exact fracture boundaries.

FIGURE 8: Mesh of 2D fractured network system with fracture extension. (a) Extension of some initial existing fractures; (b) final mesh.

total elapsed CPU time for mesh generation of this example is around 0.2 seconds.

We also considered extension of some initial existing fractures (red line segments in Figure 8(a)). The final mesh

with including initial and extended fractures can easily be constructed by the proposed local correction approach.

The second example is a three-dimensional DFN system in a cubic box with 36 circular disk fractures (see Figure 9).

FIGURE 9: Mesh of 3D fractured network system (tetrahedrons around fractures are displayed for clarity).

The final mesh includes 114,605 vertexes and 733,798 tetrahedrons. By the proposed techniques of inserting predefined nodes on fracture surfaces and local correction, all the fracture surfaces are exactly constructed. The total elapsed CPU time for mesh generation of this example is around 12 seconds, which indicates the high efficiency of the proposed approach.

6. Conclusions

Modeling and meshing of fractured network system are usually tedious and difficult due to geometric complexity of the computational region. Geometric constraints induced by existence of fractures, especially the extension of the fractures, significantly increase degree of difficulty of meshing algorithm. To exactly construct fracture boundaries, the so-called boundary recovery operation based on topological transformation is usually involved. This traditional operation relies on many complicated skills and is not easy to implement.

This paper presents an alternative and efficient approach for meshing fractured network system. The method firstly presets points on fractures and then performs Delaunay triangulation to obtain preliminary mesh by point-by-point centroid insertion algorithm. Furthermore, the fractures are exactly recovered by local correction with dynamic grid deformation approach. Smoothing algorithm is finally applied to improve the quality of mesh. To fulfill the above tasks, revised versions of dynamic grid deformation and smoothing to deal with fracture boundaries are developed. The proposed approach is efficient, easy to implement, and applicable to the cases of initial existing fractures and extension of fractures. It can exactly construct fracture boundaries without performing complicated topological transformation operations.

The proposed approach integrates some robust and efficient algorithms, including point-by-point centroid insertion, dynamic grid deformation, and mesh smoothing. Its effectiveness and high efficiency are demonstrated by a number of two- and three-dimensional examples in modeling geological problem.

Conflict of Interests

The authors do not have any conflict of interests with the content of the paper.

Acknowledgments

The authors would like to thank the support of National Basic Research Program of China (2010CB731503), the National Natural Science Foundation of China (11172004, 10772004), and Beijing Municipal Natural Science Foundation (1102020).

References

- [1] L. A. Freitag and C. Ollivier-Gooch, "A cost/benefit analysis of simplicial mesh improvement techniques as measured by solution efficiency," *International Journal of Computational Geometry & Applications*, vol. 10, no. 4, pp. 361–382, 2000.
- [2] H. Mustapha, R. Dimitrakopoulos, T. Graf, and A. Firoozabadi, "An efficient method for discretizing 3D fractured media for subsurface flow and transport simulations," *International Journal for Numerical Methods in Fluids*, vol. 67, no. 5, pp. 651–670, 2011.
- [3] D. Blessent, R. Therrien, and K. MacQuarrie, "Coupling geological and numerical models to simulate groundwater flow and contaminant transport in fractured media," *Computers and Geosciences*, vol. 35, no. 9, pp. 1897–1906, 2009.
- [4] P. L. George and H. Borouchaki, "Back to edge flips in 3 dimensions," in *Proceedings of the 12th International Meshing Roundtable*, Sandia National Laboratories, pp. 393–402, 2003.
- [5] Q. Du and D. Wang, "Constrained boundary recovery for three dimensional Delaunay triangulations," *International Journal for Numerical Methods in Engineering*, vol. 61, no. 9, pp. 1471–1500, 2004.
- [6] H. Si, "Constrained Delaunay tetrahedral mesh generation and refinement," *Finite Elements in Analysis and Design*, vol. 46, no. 1–2, pp. 33–46, 2010.
- [7] J. Liu, B. Chen, and Y. Chen, "Boundary recovery after 3D Delaunay tetrahedralization without adding extra nodes," *International Journal for Numerical Methods in Engineering*, vol. 72, no. 6, pp. 744–756, 2007.
- [8] S. Li, B. Chen, L. Zhao, and Y. Liu, "A fast mesh generation algorithm with point-by-point delaunay insertion," *Acta Scientiarum Naturalium Universitatis Pekinensis*, vol. 43, no. 3, pp. 302–306, 2007 (Chinese).
- [9] D. J. Mavriplis, "Unstructured grid techniques," *Annual Review of Fluid Mechanics*, vol. 29, no. 1, pp. 473–514, 1997.
- [10] A. Bowyer, "Computing Dirichlet tessellations," *The Computer Journal*, vol. 24, no. 2, pp. 162–166, 1981.
- [11] D. F. Watson, "Computing the n -dimensional Delaunay tessellation with application to Voronoï polytopes," *The Computer Journal*, vol. 24, no. 2, pp. 167–172, 1981.

- [12] C. L. Lawson, "Transforming triangulations," *Discrete Mathematics*, vol. 3, pp. 365–372, 1972.
- [13] H. Borouchaki and S. H. Lo, "Fast Delaunay triangulation in three dimensions," *Computer Methods in Applied Mechanics and Engineering*, vol. 128, no. 1-2, pp. 153–167, 1995.
- [14] P. L. George, "Improvements on Delaunay-based three-dimensional automatic mesh generator," *Finite Elements in Analysis and Design*, vol. 25, no. 3-4, pp. 297–317, 1997.
- [15] K. E. Thompson, "Fast and robust Delaunay tessellation in periodic domains," *International Journal for Numerical Methods in Engineering*, vol. 55, no. 11, pp. 1345–1366, 2002.
- [16] T. Kalbacher, R. Mettler, C. McDermott et al., "Geometric modelling and object-oriented software concepts applied to a heterogeneous fractured network from the Grimsel rock laboratory," *Computational Geosciences*, vol. 11, no. 1, pp. 9–26, 2007.
- [17] X. Liu, N. Qin, and H. Xia, "Fast dynamic grid deformation based on Delaunay graph mapping," *Journal of Computational Physics*, vol. 211, no. 2, pp. 405–423, 2006.
- [18] J. T. Batina, "Unsteady Euler algorithm with unstructured dynamic mesh for complex-aircraft aerodynamic analysis," *AIAA Journal*, vol. 29, no. 3, pp. 327–333, 1991.
- [19] C. Farhat, C. Degand, B. Koobus, and M. Lesoinne, "Torsional springs for two-dimensional dynamic unstructured fluid meshes," *Computer Methods in Applied Mechanics and Engineering*, vol. 163, no. 1-4, pp. 231–245, 1998.
- [20] A. A. Johnson and T. E. Tezduyar, "Simulation of multiple spheres falling in a liquid-filled tube," *Computer Methods in Applied Mechanics and Engineering*, vol. 134, no. 3-4, pp. 351–373, 1996.
- [21] T. J. Baker, "Mesh deformation and reconstruction for time evolving domains," in *Proceedings of the 15th AIAA Computational Fluid Dynamics Conference*, pp. 2001–2535, 2001.
- [22] S. Sun and B. Chen, "An algebraic deformation approach for moving grid based on barycentric coordinates," in *Proceedings of the International Conference on Computational Intelligence and Software Engineering (CiSE '10)*, pp. 1–4, Wuhan, China, December 2010.
- [23] S. L. Sun and M. W. Yuan, "High quality mesh deformation method for large scale unstructured hybrid grid," *IACM Expressions*, vol. 29, pp. 9–12, 2011.
- [24] S. L. Sun, B. Chen, J. F. Liu, and M. W. Yuan, "An efficient implementation scheme for the moving grid method based on Delaunay graph mapping," in *Proceedings of the 2nd International Symposium on Computational Mechanics (ISCM II) and the 12th International Conference on the Enhancement and Promotion of Computational Methods in Engineering and Science (EPMESC XII)*, vol. 1233, pp. 1046–1051, Hong Kong, 2009.
- [25] S. Sun and J. Liu, "An efficient optimization procedure for tetrahedral meshes by chaos search algorithm," *Journal of Computer Science and Technology*, vol. 18, no. 6, pp. 796–803, 2003.
- [26] J. Liu, S. Sun, and D. Wang, "Optimal tetrahedralization for small polyhedron: a new local transformation strategy for 3-D mesh generation and mesh improvement," *Computer Modeling in Engineering & Sciences*, vol. 14, no. 1, pp. 31–43, 2006.
- [27] L. A. Freitag and P. Plassmann, "Local optimization-based simplicial mesh untangling and improvement," *International Journal for Numerical Methods in Engineering*, vol. 49, no. 1-2, pp. 109–125, 2000.
- [28] L. Chen and J.-c. Xu, "Optimal Delaunay triangulations," *Journal of Computational Mathematics*, vol. 22, no. 2, pp. 299–308, 2004.
- [29] L. Chen, "Mesh smoothing schemes based on optimal Delaunay triangulations," in *Proceedings of the 13th International Meshing Roundtable*, pp. 109–120, Sandia National Laboratories, 2004.
- [30] P. Alliez, D. Cohen-Steiner, M. Yvinec, and M. Desbrun, "Variational tetrahedral meshing," *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 617–625, 2005.
- [31] S. Sun, H. Bao, and M. Liu, "Smoothing algorithm for tetrahedral meshes by error-based quality metric," in *Proceedings of the International Conference on Computational Intelligence and Software Engineering (CiSE '10)*, pp. 1–4, Wuhan, China, December 2010.